Why are "What" and "Where" Processed by Separate Cortical Visual Systems? A Computational Investigation

Why are "What" and "Where" Processed by Separate Cortical Visual Systems? A Computational Investigation

Jay G. Rueckl
Harvard University
Kyle R. Cave
Massachusetts Institute of Technology
Stephen M. Kosslyn
Harvard Universtiy

Abstract

In the primate visual system, the identification of objects and the processing of spatial information are accomplished by different cortical pathways. The computational properties of this "two-systems" design were explored by constructing simplifying connectionist models. The models were designed to simultaneously classify and locate shapes that could appear in multiple positions in a matrix, and the ease of forming representations of the two kinds of information was measured. Some networks were designed so that all hidden nodes projected to all output nodes, whereas others had the hidden nodes split into two groups, with some projecting to the output nodes that registered shape identity and the remainder projecting to the output nodes that registered location. The simulations revealed that splitting processing into separate streams for identifying and locating a shape led to better performance only under some circumstances. Provided that enough computational resources were available in both streams, split networks were able to develop more efficient internal representations, as revealed by detailed analyses of the patterns of weights between connections.

Introduction

The performance of the primate visual system is remarkably robust under a wide variety of circumstances. Not only can we recognize objects when they are at different distances and in different lighting, but we also can recognize objects when they are seen from different angles of regard, projecting different images onto the retina and into the brain. Indeed, the simple fact that we can recognize objects when they are in different places in the visual field is remarkable, given the differences in the images being projected on the retina. Somehow these different images, resulting in different physiological states in the brain, are able to access the same stored information in memory. In this article we explore different computational means of achieving what is sometimes called "stimulus equivalence across retinal translation" (Gross & Mishkin, 1977), namely the ability to recognize an object when its image strikes different parts of the retina.

A variety of computational mechanisms have been proposed to account for our ability to recognize objects in different parts of the visual field. We can characterize these alternatives as falling along a dimension. At one extreme, a separate representation of an object would be associated with each distinct location in the visual field. Thus, no matter where the image fell, it would be associated with a representation stored in memory. At the other extreme, only a single representation might be stored in memory, with the input being normalized so that a stored representation can be matched against images at any retinal location. One way to accomplish this is to represent the parts of an object relative to the object itself (that is, to use object-centered coordinates), as suggested by Feldman (1985), Hinton (1981), Marr (1982), and others. Various intermediate possibilities also exist. For example, there might be not one but two stored representations (perhaps one for each visual hemifield), or four (one per quadrant), and so on until the limit is reached.

The advantages of storing fewer representations are obvious, but these advantages are not without their costs.

If one has only a small number of stored representations, which register input over a wide range of positions, the object identification system will lose information about spatial location. Gross and Mishkin (1977) noticed this, and pointed out that if this kind of mechanism exists, a second mechanism must be employed in order to keep track of spatial relationships. And in fact a variety of evidence suggests that the primate visual system processes object properties and location information separately. The object identification system presumably employs position-invariant representations, and a second system is responsible for registering spatial information. Much of the evidence supporting this view has been summarized by Ungerleider and Mishkin (1982; see also Van Essen, 1985).

Although there is now an impressive accumulation of neuroanatomical and neurophysiological evidence for the separation of processing of "what" and "where," the analysis of why the brain evolved these separate pathways is fundamentally computational. That is, the argument offered by Gross and Mishkin (1977) rests on considerations of the kinds of information processing that can be performed by such a system. Gross and Mishkin argued at an informal level, however, and it is not always clear how to evaluate such arguments.

The present article is an investigation of the computational demands on a single system that represents both shape and location versus two systems, with the two kinds of information being represented separately. We will investigate the computational issues that arise when considering the two designs, and later consider possible implications of our findings for the brain.

The hypothesis considered here is that the visual system can represent identity and location information more readily if separate processing resources are dedicated to each of these tasks. In order to explore this possibility, we will examine how easily different types of systems can form appropriate internal representations. The type of system we will explore has been called, variously, a "connectionist" architecture, a "parallel distributed processing" system, or a "neural network" (e.g., McClelland & Rumelhart, 1986; Rumelhart & McClelland, 1986). In a connectionist model, processing occurs in a highly interconnected network of simple processing units called nodes. Nodes communicate by sending excitatory and inhibitory signals over weighted connections (the greater the absolute weight, the stronger the excitation or inhibition), and each node determines its activation value as a function of the signals it receives from other nodes. The inputs and outputs of a computation are represented by patterns of activity over specified sets of nodes. These representations are "distributed" in that each representation involves many nodes, and each node is involved in many representations.

Although these connectionist networks do not correspond to real neural nets (for a host of reasons; e.g., see Rumelhart and McClelland, 1986, chapter 4), they do seem to capture some important properties of how actual neural networks operate. Indeed, Zipser and Andersen (1988) found that the kind of networks we will explore here provided surprisingly good models of the behavior of certain neurons in area 7a of the macaque brain. Thus, although the network is not a literal model of the relations among neurons, it can be taken to reflect more abstract computational properties of processing in the brain (see Sejnowski, Koch, & Churchland, 1988).

The class of connectionist systems we explore here involves three non-overlapping sets of nodes called *layers*. Nodes in the input layer receive input from the environment, and send excitatory and inhibitory signals to nodes in the hidden layer, which in turn send signals to nodes in the output layer. Each layer plays a distinct role in the computation performed. The input and output layers are used to represent the inputs and outputs of the system, respectively. The hidden layer is used to create an internal representation of the input that extracts information needed to map the input onto the appropriate output. Although the hidden layer is not always necessary, there are a wide variety of functions that cannot be computed without it (see Rumelhart, Hinton, & Williams, 1986).

The output of the system in response to an input is determined by the pattern of connectivity among the nodes and the pattern of weights on the connections. One attractive aspect of connectionist models is that a number of simple learning algorithms have been devised for finding an appropriate pattern of weights. The algorithm used here (called the *generalized delta rule*) compares the actual output of the system with the appropriate output given the input, and modifies each connection strength (weight) as a function of the discrepancy between these patterns. Over time, the system tends to converge on a pattern of weights that produces the appropriate output for each input.

In the present context, the value of this learning algorithm is that it provides a tool for examining the relative merits of the one-system and two-system approaches to representing object identity and location. We make no claims about the neurological validity of the learning algorithm or, for that matter, about learning at all. We use performance accuracy as a convenient measure of how difficult it is for a given system to form the appropriate internal representations. Our hypothesis is that, all else being equal, a system will form appropriate representations more easily if it devotes separate resources to computing "what" and "where" than if it does not. In order to test this hypothesis, connectionist models of the one-system and two-system mechanisms were simulated. These models were confronted with simplified versions of a task performed by the visual system. Specifically, the models were presented with a number of shapes that could appear at various locations on a "retinal" grid, and were required to classify both the identity and the location of each shape. The accuracy of performance was taken to indicate the ease with which identity and location information could be represented in the system.

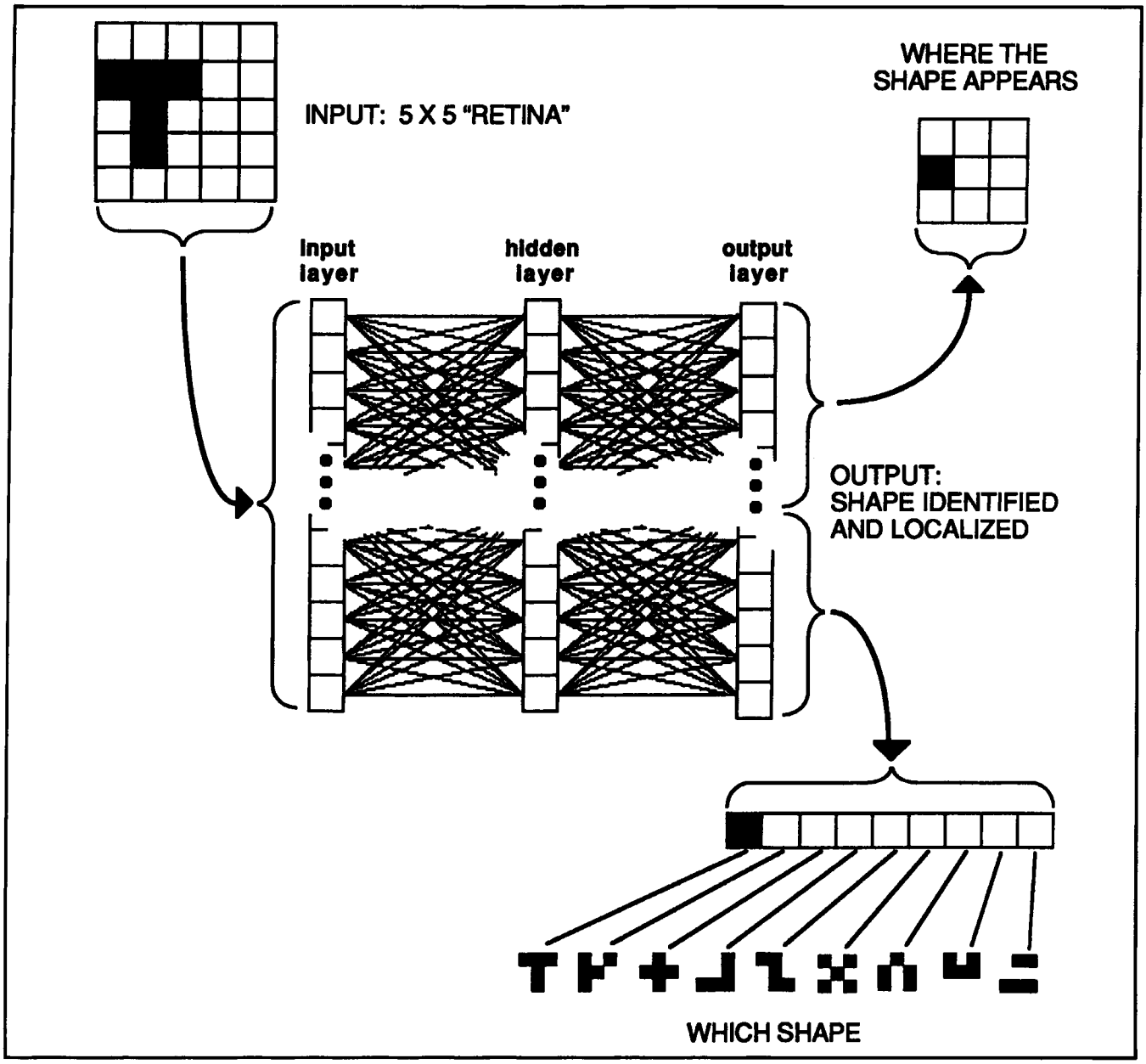

Figure 1. The unsplit model.

Structure of the Models

The models differed only in the degree to which separate resources were dedicated to computing identity and location. The unsplit model, a one-system mechanism, was a feed-forward network with 25 input nodes, 18 hidden nodes, and 18 output nodes. As is illustrated in Figure 1, the 25 input nodes were organized into a 5 x 5 matrix, and patterns could be represented by selectively activating specific input nodes. Each hidden node received input from all 25 input nodes, and each hidden node in turn sent input to all 18 output nodes. Nine shapes were formed by activating different combinations of input nodes, as is illustrated in Figure 1. These shapes were defined as different combinations of cells within a 3 x 3 grid. The 3 x 3 grid for each shape could be centered at any of nine different locations on the 5 x 5 "retinal" input grid. Thus, there were 81 different combinations of shape and location, and each combination was represented by a binary pattern such that a node was turned on if its corresponding cell in the input array was covered by the shape and turned off otherwise.

The 18 output nodes were divided into two subsets of nine nodes each. Nodes in the "what" subset were responsible for indicating the identity of the input. Each shape was associated with one of the nine "what" nodes, and the system was considered to have recognized a presented shape if it turned on the "what" node associated with that shape and turned off the "what" nodes associated with the other shapes. Similarly, each of the nine locations at which a shape could be presented was associated with

one of the nine "where" nodes, and the system was considered to have correctly located an input if it turned on the "where" node associated with that location and turned off the others.

The two-system mechanism was embodied by the split model. The structure of the split model differed from that of the unsplit model only in the pattern of connectivity between the hidden and output layers. In the unsplit model the nodes in the hidden and output layers were fully interconnected; that is, each hidden node was connected to all of the output nodes, and each output node received a connection from each of the hidden nodes. In the split model, on the other hand, the pattern of connectivity between the hidden and output layers was restricted by partitioning the hidden nodes into two subsets of nine nodes each. Nodes in one of the subsets were connected only to the "where" output nodes (constituting the "where" system), whereas nodes in the other subset of the hidden layer were connected only to the "what" output nodes (constituting the "what" system).

Thus, in the split model, each hidden node sent input to only half of the output nodes, and each output node received input from only half of the hidden nodes. The split model thus functioned as two distinct systems, overlapping only at the input level. For the interested reader, additional details about the method are included at the end of this article.

Experiment 1

The first experiment examined the performance of two networks, which differed only in whether the hidden nodes were organized into one (the unsplit model) or two (the split model) groups. The network was presented with randomized sets of trials in which each shape was presented in each location. The network received feedback after each trial regarding its accuracy, and the learning procedure described by Rumelhart et al. (1986) was used to adjust the weights among the connections. The degree of improvement in the network's responses was noted as increasing numbers of trials were presented, and this measure was used to compare how well the two networks formed the requisite internal representations.

Results and Discussion

The results are presented in Figure 2. In this figure the summed squared error (a measure of the discrepancy between the actual and correct responses to a given input) for the split and unsplit models is plotted as a function of blocks of trials (a block was a complete set of 81 trials in which each shape appeared at each location). The values plotted in this figure are means from eight different networks, each starting with a different random configuration of weights. As can be seen in the figure, the performance of both models improved over blocks, with most of the learning occurring in the first 100 blocks. However, contrary to our hypothesis the unsplit model

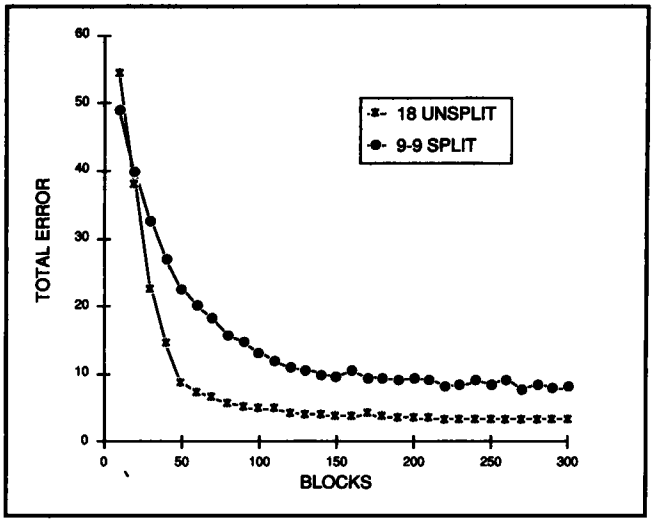

Figure 2. The summed squared error for the split and unsplit models in Exp. 1 as a function of blocks of trials.

actually learned more quickly and efficiently than did the split model! This result is evident both in the early blocks, where the total error decreased more rapidly for the unsplit model, and in the later blocks, where there is consistently more error for the split model. After 300 blocks of learning, the unsplit net is clearly producing less error, $F(1,14) = 24.11, p < .001$.

Although these results were surprising, further inspection suggested an explanation. Figures 3 and 4 present the results broken down by the two types of outputs. In Figure 3 error is again plotted as a function of blocks. In this case, however, only the error across the "what" output nodes has been summed. (Recall that the number of "what" output nodes is the same for the split and unsplit models.) As is evident, the pattern of results for the "what" nodes resembles the overall pattern displayed in Figure 2. Again there is a consistent and substantial advantage for the unsplit model. In contrast, the results for the "where" nodes, presented in Figure 4, do not resemble the overall results. In this case there is little difference between the models, and both models perform virtually perfectly after a small number of trials.

Why should the split model do as well as the unsplit model in classifying location, but do worse in classifying identity? One possibility is that the two classifications are not equivalent in difficulty. In general, the difficulty of an input/output mapping decreases as a function of the systematicity of that mapping (i.e., the degree to which similar input patterns are mapped onto similar output patterns and dissimilar input patterns are mapped onto dissimilar output patterns). In the case of classifying location, the level of systematicity is relatively high. Each output response is associated with patterns that are relatively similar in that these patterns involve different combinations drawn from the same set of nine input nodes. Thus, all of the input patterns associated with a given output have highly overlapping subsets of activated input

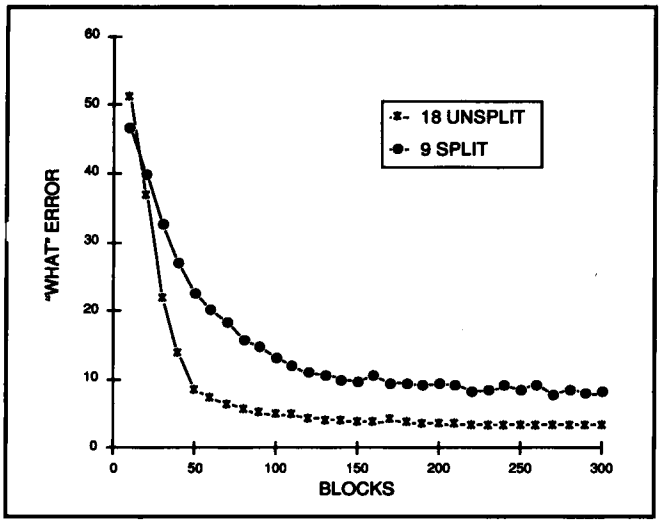

Figure 3. The "what" error for the split and unsplit models in Exp. 1 as a function of blocks of trials.

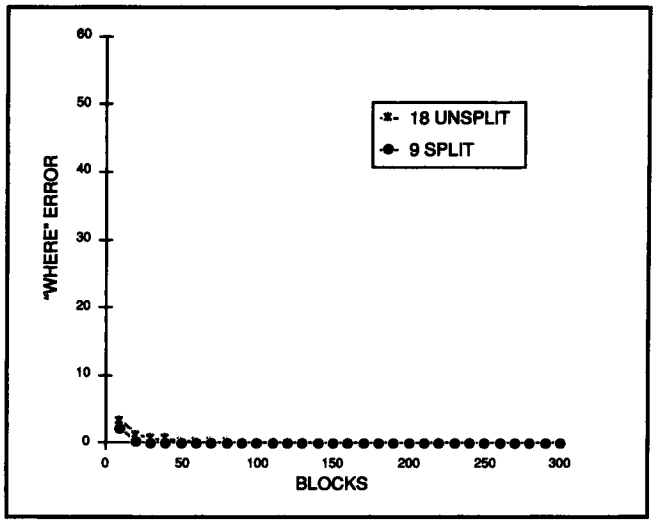

Figure 4. The "where" error for the split and unsplit models in Exp. 1 as a function of blocks of trials.

nodes. Conversely, input patterns with little or no overlap tend to be mapped onto different "where" outputs.

In classifying shape identity, on the other hand, the degree of systematicity is relatively low. Because the same shape typically will turn on different input nodes as it moves around on the input matrix, quite different input patterns will be mapped onto the same "what" output. Furthermore, because the most similar input patterns (from the system's point of view) tend to be those involving different shapes appearing at the same location in the input array, "similar" input patterns tend to be mapped onto different "what" outputs.

The argument that the "what" classification is considerably more difficult than the "where" classification is supported by the results of Experiment 1. Note that for both networks, the total error per block is greater for the "what" output nodes than for the "where" output nodes. Given this finding, we can offer one possible explanation of why the split model classified location as well as the unsplit model but classified identity worse than the unsplit model: the hidden nodes may not have been allocated properly in the split model. That is, although nine dedicated hidden nodes may have been more than enough to classify location, the same number of nodes may not have been sufficient for classifying identity. Thus, the failure of the split model to outperform the unsplit model may have been due to our forcing an inefficient allocation of hidden nodes to the two types of computations.

Experiment 2

In the second experiment we manipulated the number of hidden nodes that were allocated to computing identity and location in split networks. If the results of Experiment 1 occurred because identity is more difficult to compute than location, then performance in split networks should improve when proportionally more nodes are allocated to computing identity.

Three different split models were simulated in this Experiment. In the 12-6 split model, 12 hidden nodes were allocated to project only to the "what" (identity) output nodes and the remaining 6 nodes were allocated to project only to the "where" (location) output nodes. Similarly, in the 14-4 and 15-3 split models, 14 and 15 hidden nodes were allocated to the "what" system, respectively, with the remaining nodes allocated to the "where" system. Each model was tested in eight simulations of 300 blocks each, with each block again including 81 trials (all combinations of shapes and locations). Other than the change in the number of hidden nodes assigned to each system, the method and procedure used in these simulations was identical to that used in Experiment 1.

Results and Discussion

The results of Experiment 2 are presented in Figure 5. In this figure the summed squared error for each model is displayed as a function of blocks. For purposes of comparison, learning curves for the unsplit and 9-9 split models from Experiment 1 are also presented. The results demonstrate that the performance of the split model depends critically on the proportion of hidden nodes dedicated to computing identity and location. The 9-9 and 15-3 split models performed relatively poorly compared to the unsplit model. The 14-4 and 12-6 split models, on the other hand, attained a better level of performance than did the unsplit model. A Newman-Keuls test comparing the best of the split models (14-4) against the unsplit model revealed that the split model produced significantly less error after 300 blocks of learning, $p < .01$.

Examination of the error for the computation of identity and location considered separately reveals why the 9-9 and 15-3 split models fared worse than the unsplit model. Figure 6 presents the "what" error for each of the models. As is apparent in the figure, the "what" error for

the split models was systematically related to the number of hidden nodes dedicated to computing this information: the more hidden nodes, the better the asymptotic performance and the more quickly the system reached that level of performance. As is evident in Figure 7, the same pattern emerged for the computation of location: the more nodes dedicated to that computation, the better the performance and the more quickly the system learned. These results suggest that the 9-9 and 15-3 split models were inferior to the unsplit model because too few nodes were allocated to the "what" and "where" systems, respectively. However, when both the "what" and "where" systems were allocated a sufficient number of nodes, the split model outperformed the unsplit model.

Note, however, that although the asymptotic performance of the best split models surpassed that of the unsplit model, the performance of even these split models lagged behind that of the unsplit model early in the learning sequence. This initial advantage of the unsplit model was probably due to the fact that it had twice as many connections between the hidden layer and the output layer than did the split models. This difference in the number of connections may have influenced the results in several ways. One possibility is that the greater number of connections may have allowed the unsplit model to formulate easily a useful approximate pattern of weights. That is, given the greater number of connections, the unsplit model had a better chance of initially adjusting weights on some subset of those connections that provided an approximate mapping.[1]

For both split and unsplit models it is clear that performance depends critically on the number of hidden units. With too few hidden nodes, neither model can adequately represent the location and identity of the stimulus. With too many hidden nodes, even an inefficient coding scheme will produce little error, and because of a floor effect, both split and unsplit models will produce error levels near zero. This was evident in a comparison of an unsplit net with 24 hidden nodes with a split net with 18 what hidden nodes and 6 where hidden nodes. The mean error rate for the split net was lower, just as it was in the models described above (.000019 for the split vs. .50 for the unsplit), although in this case the difference did not reach significance. We suspect that as the number of hidden nodes increases beyond 24, the difference between split and unsplit error will diminish further as both approach zero.

Types of Hidden Nodes: Receptive and Projective Fields

In the models the hidden nodes are used to recode the input in a way that allows each network to compute both identity and location. In the split networks, two distinct internal representations are used. The pattern of activation over the hidden nodes that project to "what" output nodes is an internal representation used to compute identity, whereas the pattern of activation over the hidden nodes

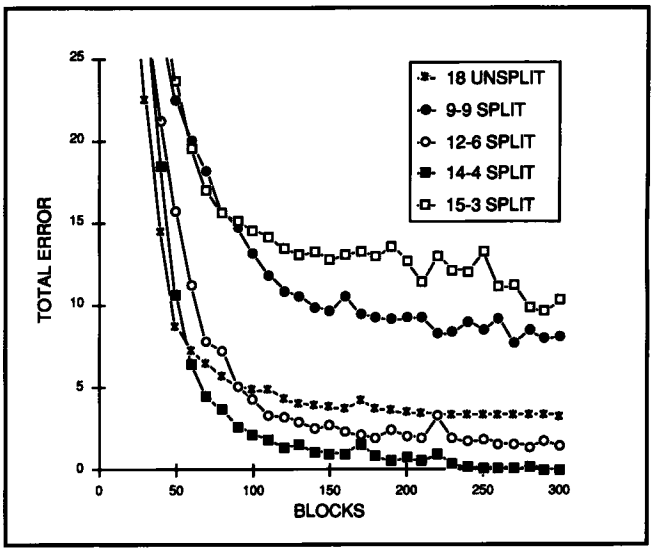

Figure 5. The summed squared error for the split and unsplit models in Exp. 2 as a function of blocks of trials.

that project to the "where" output nodes is an internal representation used to compute location. In the unsplit model, on the other hand, a single internal representation is created. Because each of the hidden nodes in the unsplit model is connected to all the output nodes, each hidden node must carry information that is relevant to both computations.

If the hidden nodes are indeed performing different tasks in the split and unsplit models, one might expect the hidden nodes to recode the input in qualitatively different ways in the two kinds of networks. One way to investigate this possibility is to examine the strengths of the connections coming to and leading from the hidden nodes. By looking at the pattern of connections coming to a hidden node from the input nodes (i.e., its "receptive field"), one can discover the regularities in the input to which that hidden node was sensitive. Similarly, by looking at the pattern of connections leading from a hidden node to the output nodes (its "projective field"), one gets information about which output patterns that node helped create (cf., Lehky & Sejnowski, 1988).

Split Networks: "What" Hidden Nodes: An examination of the hidden nodes revealed a number of regularities in their patterns of connections. Several of the more noticeable types of patterns are illustrated in Figure 8. In this figure, each 5 x 5 array represents the receptive field of a hidden node. Each square in a receptive field array represents the weight for the connection between the hidden node and one input node in the "retinal" matrix. Darker shadings represent more strongly negative (inhibitory) weights, whereas lighter shadings represent more strongly positive (excitatory) weights. In addition, the column graph to the right of each 5 x 5 array displays the weights for the connections from that hidden node to the "what" output nodes. Along the top are the nine shapes

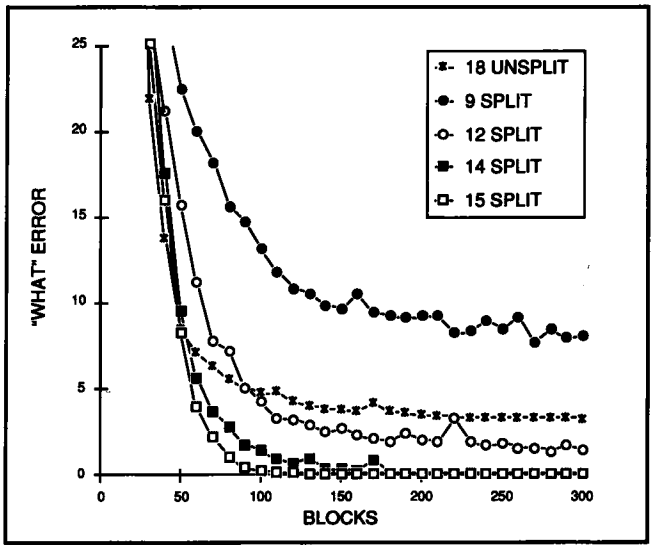

Figure 6. The "what" error for the split and unsplit models in Exp. 2 as a function of blocks of trials.

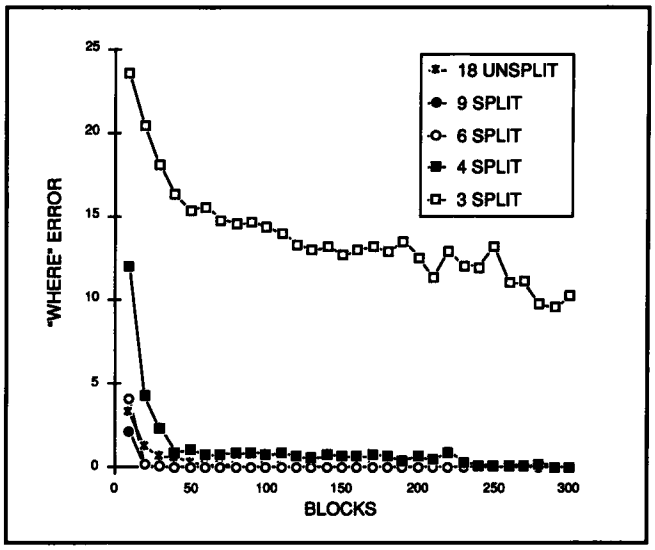

Figure 7. The "where" error for the split and unsplit models in Exp. 2 as a function of blocks of trials.

used as input to the system, each of which was represented by a single output node. (Recall that these shapes are actually defined as sets of pixels in a 3 x 3 array.) Columns extending up represent positive weights on the connections to the output nodes for the corresponding shapes, whereas columns extending down represent negative weights to those output nodes.

One striking class of "what" hidden nodes had receptive fields organized in alternating stripes of excitatory and inhibitory connections (Figure 8a, b, c). The orientation of the stripes varied: some were arranged horizontally, others were arranged vertically, and still others were arranged diagonally. When the pattern of stripes extends across the entire input array, as it usually did, then the hidden node can provide useful information about the stimulus shape no matter where it appears (although of course it is almost useless for computing the location). We found that these nodes would often appear in complementary pairs within a network. For example, a node with a receptive field of excitatory and inhibitory vertical stripes was typically accompanied by another node that had positive input connections wherever the first one had negative connections and vice versa. These nodes tended to have relatively strong positive or negative connections to most or all of the output nodes, and thus played a role in the identification of most of the shapes.

A second class of "what" hidden nodes received positive inputs from input nodes along one or two borders of the input array (Figure 8d, e, f). Some of these nodes received input from a single row of input nodes along the border, whereas others received input from two adjacent rows along the border. These hidden nodes responded to shapes with a number of "on" pixels along one edge. Of course, these hidden nodes are only useful when the shape appears at the appropriate border of the input array: it will only be possible to identify those shapes that are "heavy on top" when they appear at the top of the input array. However, because the input array is small, every position except one is along one of the borders, and almost half are along two. Thus, although these nodes will not be able to provide useful information as often as the stripe detectors, with a small input array they will often be helpful in identifying shapes.

A third class of hidden nodes had receptive fields with a single stripe of excitatory connections surrounded on both sides by inhibitory regions (Figure 8g). As was the case for the "stripe nodes" described above, the orientation of the stripes in the receptive fields of these nodes also varied: some nodes had a horizontally oriented stripe, whereas others had a vertically oriented stripe. These nodes seemed to be responsible for detecting line segments of the proper orientation—particularly when these line segments were in the middle column or row of the 3 x 3 letter grid.

Not all of the "what" hidden nodes could be classified as "stripe", " single stripe," or "border" nodes. Many of the nodes that were not organized by stripes or borders generally seemed to be specialized for particular features at a particular position. These nodes (which we will refer to as "ad hoc nodes") had irregularly shaped receptive fields (e.g., Figure 8h) and sometimes had projective fields with strong positive connections to one or two output nodes and weak or moderately negative connections to the rest. Finally, a few nodes had seemingly random receptive fields and were not strongly connected to any of the output nodes. Because these nodes had little influence on the behavior of the system, we will refer to these nodes as "unused." We shortly will present statistics comparing the frequency of these different types of nodes in the different models.

Split Networks: "Where" Hidden Nodes: A number of

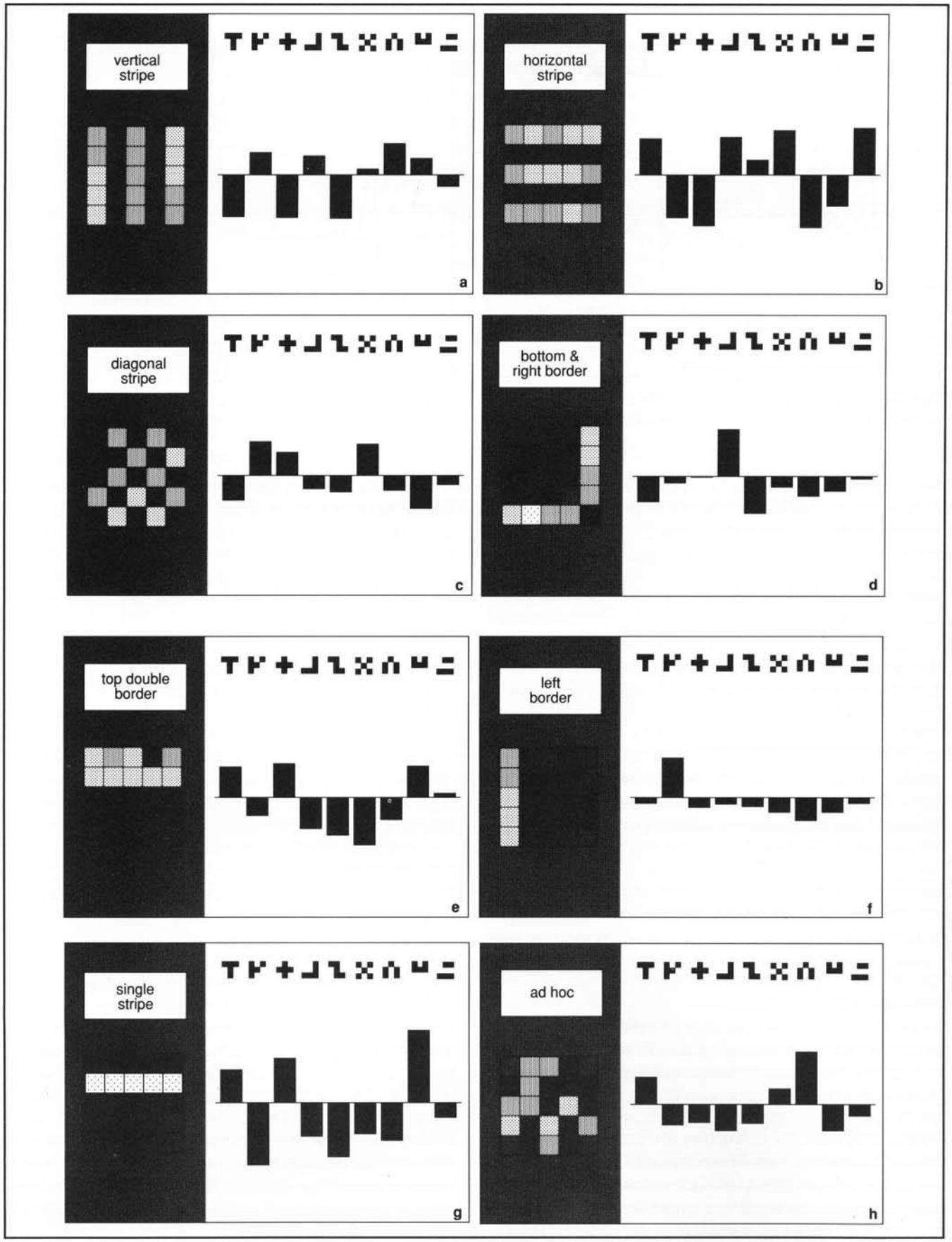

Figure 8. The receptive and projective fields of "what" hidden nodes: (a,b,c) "stripe" nodes; (d,e,f) "border" nodes; (g) a "single stripe" node: (h) an "ad hoc" node.

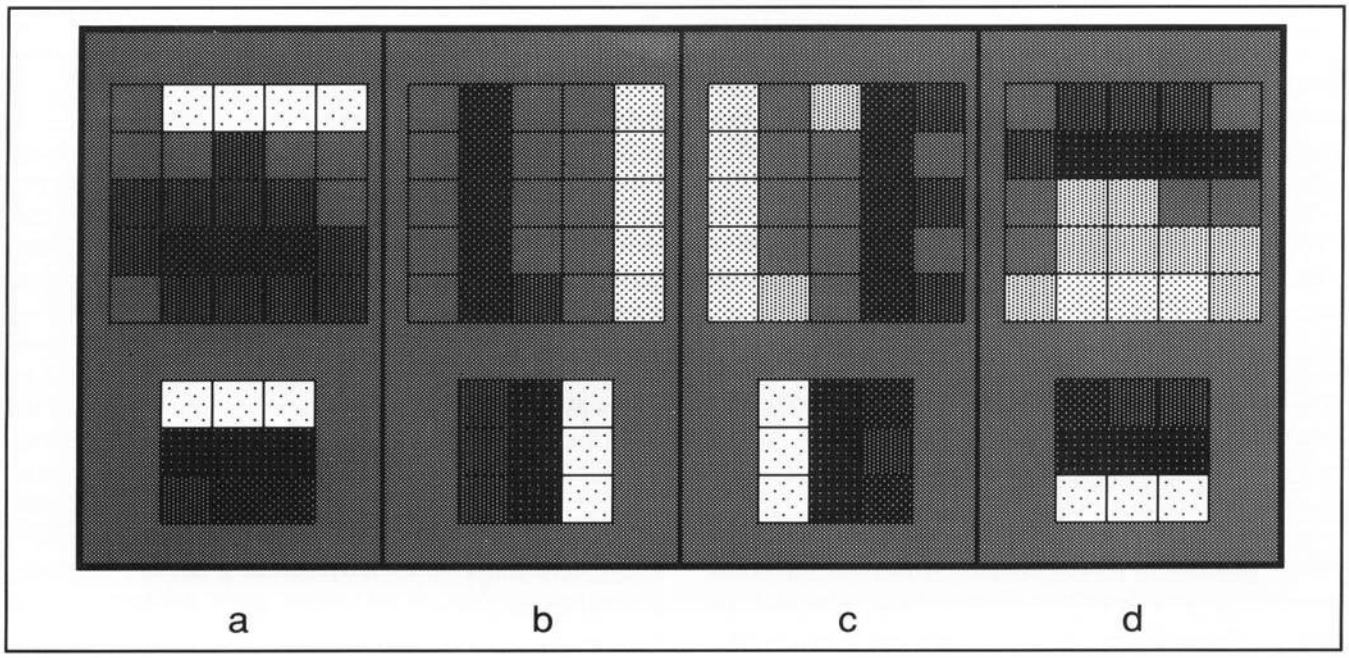

Figure 9. The receptive (top) and projective (bottom) fields of the "where" hidden nodes from one run of the 14-4 split model: (a,b,c) "border" nodes; (d) a "contiguous" node.

regularities also emerged from the patterns of connections to and from the "where" hidden nodes. Figure 9 illustrates the input and output connections for some of these nodes. As was the case in the previous figure, each 5 x 5 array represents the receptive field of a hidden node. Underneath the receptive field is a 3 x 3 array illustrating the node's projective field—the weights between the hidden node and the nine "where" output nodes. Two of the major categories of "where" hidden nodes are represented in Figure 9. One category included nodes with receptive fields like those of the "border" nodes described above (Figure 9a, b, c). The second category illustrated in Figure 9 included nodes with receptive fields containing contiguous excitatory regions covering areas other than one or two borders (i.e., extending into the interior of the "retinal" grid; see Figure 9d). Some "where" nodes did not resemble members of either of these categories. These "ad hoc" "where" nodes generally had receptive fields with several noncontiguous excitatory areas, as well as excitatory connections to output nodes representing noncontiguous locations.

The "where" nodes shown in Figure 9 represent the entire ensemble of "where" nodes from one 14-4 split network, and nicely illustrate the sort of encoding scheme adopted by these networks to compute location. (However, the receptive fields were not always contiguous in these networks.) In every 14-4 network, each of the "where" nodes responded to a small number of positions, and there was always a unique combination of nodes responding to each location. This coding scheme makes clear why split networks with three "where" hidden nodes perform so much more poorly than those with four. With three nodes, there are only eight unique possible combinations, and thus it is not possible to represent all nine positions.

Adding a fourth "where" node brings the "where" error down substantially, as shown by a Newman-Keuls test of the error level after 300 blocks of learning, $p < .01$. Figure 7 illustrates that adding more "where" nodes beyond four decreases the "where" error level even further, but only slightly. This is because the additional "where" nodes result in a less efficient representation, with many of the hidden nodes responding to only a single position. Thus, these networks require four nodes to represent location adequately, but any more than four provide little additional benefit.

Unsplit Networks: Hidden Nodes: Two aspects of the hidden nodes in the unsplit networks are particularly noteworthy. First, most of these nodes had strong connections to both "what" and "where" output nodes. Thus, it was not the case that some of these nodes were dedicated exclusively to the computation of location whereas others were dedicated exclusively to the computation of shape. Instead, the vast majority of unsplit hidden nodes participated in both computations. This observation was quantified by considering the correlation between the sums of the absolute values of the weights to the "what" and "where" output nodes. If the hidden nodes were specialized for "what" and "where," then those with strong weights to the "what" output nodes should have weak weights to the "where" output nodes, and vice versa. Thus, in this case the correlation between the strength of the "what" and "where" connections across all the unsplit hidden nodes should approach –1.00. However, the actual correlation between the "what" and "where" values

was −.31, indicating that, at most, a small amount of specialization occurs.

The second important observation about the hidden nodes in the unsplit model concerns the relative frequency of different types of nodes. That is, the unsplit hidden nodes showed many of the same characteristics of the nodes in the split models, with some nodes displaying horizontal, vertical, and diagonal stripes in their receptive fields, others with contiguous excitatory regions, and still others that responded to the pixels along a border of the input array. However, the relative frequency of these types of nodes differed from that of the split model. In order to quantify this observation, two persons (the first and second authors) independently classified the hidden nodes from each of the unsplit networks and the 14-4 and 9-9 split networks. Nodes were classified as "stripe" nodes, "border" nodes, "contiguous" nodes, or "single stripe" nodes if they had receptive and projective field properties as was described above. Nodes that did not meet the criteria for any of these categories were labeled "ad hoc" nodes if they had moderately strong connections to at least some of the output nodes, and "unused" if all of their output connections were weak.

Table 1 presents the results of this analysis. The values in this table represent the proportion of hidden nodes (across eight simulations of each model) that fell into each category. As is clear from the table, the distribution of nodes of the various types differed across models. In the split models, the "where" system included mainly "contiguous," "border," and "ad hoc" nodes, whereas the "what" system included roughly equal numbers of "stripe," "border," and "ad hoc" nodes. In the unsplit model, most of the nodes fell into the "contiguous" or "ad hoc" categories, and fewer nodes fell into the "stripe" and "border" categories. Based on the distribution of nodes in the split model, the "contiguous" nodes seem specialized for providing location information per se. Thus, somewhat surprisingly, the relatively high frequency of "contiguous" nodes in the unsplit model, along with the relatively low frequency of "stripe" and "border" nodes, suggests that the solution found by the unsplit model is influenced more by the constraints of the location process than by the constraints of the identification process.

To summarize, examination of the hidden nodes of the split networks revealed that the "what" and "where" hidden nodes have very different receptive fields, but each kind of receptive field is clearly sensible given the task being performed. Many hidden nodes of the unsplit networks have receptive field properties similar to nodes found in the split networks. However, the distribution of the various types of nodes differed in the two classes of models. In addition, the projective fields of the nodes in the unsplit networks differed from those in the split networks in that they almost always included strong connections to both "what" and "where" output nodes. Thus, the hidden nodes in the unsplit networks do double duty, helping to compute both identity and location, and they do this by recoding the input in a way that differs from the recoding scheme used by the split networks.

Analyses of Overall Network Representations

Although classifying the types of connections to and from individual hidden nodes is interesting and often very illuminating, such taxonomies have limited value. The various hidden nodes are working together, and it is the pattern of activity defined over the entire ensemble that corresponds to the internal representation. In order to examine such overall internal representations we have quantitatively analyzed the patterns of weights formed in the networks using multidimensional scaling. As will be described in this section, these results provide further support for our inference that the hidden nodes function differently in the split and unsplit models, and provide additional insights into the nature of these differences.

The assumption underlying this analysis is that the pattern of weights to an output node depends on the coding scheme used by the hidden nodes to represent the input. If two stimuli tend to be represented by relatively similar patterns over the hidden nodes, then the patterns of weights from the hidden nodes to the output nodes representing those stimuli should also be relatively similar. Conversely, if two stimuli tend to be represented by relatively dissimilar patterns over the hidden nodes, then the patterns of weights from the hidden nodes to the output nodes representing those stimuli should be relatively dissimilar. Thus, if the split and unsplit networks develop the same coding scheme, pairs of output nodes that have relatively similar patterns of connectivity in the split model should also have relatively similar patterns of activity in the unsplit model.

The similarity of the patterns of connectivity to two output nodes was evaluated by correlating the weights on each of the connections to one output node with the weights on the corresponding connections to the other output node. For a given split or unsplit network, a correlation matrix can be formed by evaluating the similarity of the weights to each pair of output nodes. Correlation matrices of this sort were produced for each of the "what" output nodes from the eight simulations of the unsplit network in Experiment 1 and from each of the eight simulations of the 14-4 split network in Experiment 2. These correlation matrices were then individually submitted to a nonmetric multidimensional scaling analysis using Kruskal's stress formula and a Euclidean distance metric as implemented in SYSTAT 3.2 for the Macintosh. Thus, we produced a total of sixteen different solutions. (Only the "what" nodes were analyzed, both because the performance of the split and unsplit networks differed the most on the "what" task and because the coding scheme used to compute location can be easily understood from an examination of the patterns of weights to and from the hidden nodes.) The mean stress was 0.168 for the unsplit solutions, and 0.174 for the split solutions.

Representative solutions for the split and unsplit

Table 1

Category	14-4 "what"	14-4 "where"	9-9 "what"	9-9 "where"	Unsplit
Stripe	.32	.00	.31	.00	.12
Single Stripe	.04	.05	.12	.00	.01
Border	.25	.27	.24	.20	.09
Contiguous	.01	.44	.01	.65	.35
Ad hoc	.32	.25	.31	.15	.43
Unused	.05	.00	.00	.00	.00

The distribution of hidden nodes in the 14-4 split, 9-9 split, and unsplit models. Note. Values are the proportion of nodes of each type across eight simulations of each model. Inter-rater agreement was 84.3%.

networks are presented in Figures 10 and 11, respectively. In order to interpret these solutions, we grouped the items on the basis of two sorts of attributes. One class of groupings stemmed from the observation that the receptive fields of some hidden nodes were organized into horizontal, vertical, or diagonal stripes. Thus, groups of neighboring shapes that elicited strong responses from each of these types of nodes were identified. In order to determine whether a shape would elicit a strong response from a given type of node, it was first positioned in the receptive field so that the maximum number of pixels fell on the excitatory regions of the node's receptive field. If there were fewer than two pixels that fell on the inhibitory regions of the receptive field, the shape was then classified as one that would elicit a strong response from nodes of that type.

The second class of groupings stemmed from the observation that other hidden nodes (the "contiguous," "border," and "single stripe" nodes) had receptive fields organized into contiguous excitatory and inhibitory regions. These nodes are particularly sensitive to the distribution of "on" pixels within the 3 x 3 grid. For example, if a receptive field contains a weak inhibitory region and a strong excitatory region, a shape in a location straddling the boundary between these regions may fail to excite the node if only one "on" pixel falls on the excitatory region, but may evoke a response if two "on" pixels fall in this region. Thus, shapes were also classified by the distribution of "on" pixels in the 3 x 3 grid. Some of the groupings were based on properties of the "border" nodes. For example, if a shape had two or three "on" pixels in the top row of the 3 x 3 grid, it was classified as "heavy on top." Shapes were also classified as "heavy on bottom," "heavy on left," and "heavy on right." Other groupings were based on properties of the "single stripe" nodes. Shapes with three "on" pixels lined up horizontally were classified as "single horizontal." Corresponding classifications were made for "single vertical" and "single diagonal." Figure 12 provides a summary of the classification scheme.

In each of the two-dimensional scaling solutions for the "what" output nodes, we were able to group the nine output nodes according to the classifications in Figure 12. A group was formed whenever all the shapes in a particular category (or the complement of that set) appeared in the same area with no other shapes intermixed. Table 2 presents the frequency with which the configurations within an MDS solution could be grouped according to these classifications. The data in Table 2 suggest that the stripe patterns play a more important role in split networks whereas border and single-stripe patterns are more important in unsplit networks. A total of 18 stripe groups were found in the scaling solutions for the split nets, but only 11 stripe groups were found in the unsplit net solutions. In contrast, the unsplit nets produced 38 border and single stripe groupings, whereas the split nets produced only 31. The difference in the distribution of groupings is consistent with the claim that the hidden nodes in the split and unsplit models recode the input in different ways.

We further explored how the shape properties listed in Figure 12 influenced the structure of the split and unsplit networks by testing how well these properties predicted the values in the correlation matrices used in the MDS scalings above. For each pair of input shapes, we calculated a similarity measure by finding the proportion of shape properties that both shapes shared or lacked. These measures were calculated separately for the stripe properties, which can be detected with position-independent feature detectors, and for the border and single stripe properties, which are position-dependent. For instance, shapes D and E (see Figure 12) both elicit strong responses in horizontal stripe nodes, and neither elicits strong responses in diagonal stripe nodes. However, they differ in the vertical stripe response. Because they have the same value for two of the three stripe properties, this pair of shapes has a stripe similarity measure of 2/3.

These similarity measures were compared with the correlation matrices used in the MDS scalings, with separate comparisons for split and unsplit models and for stripe

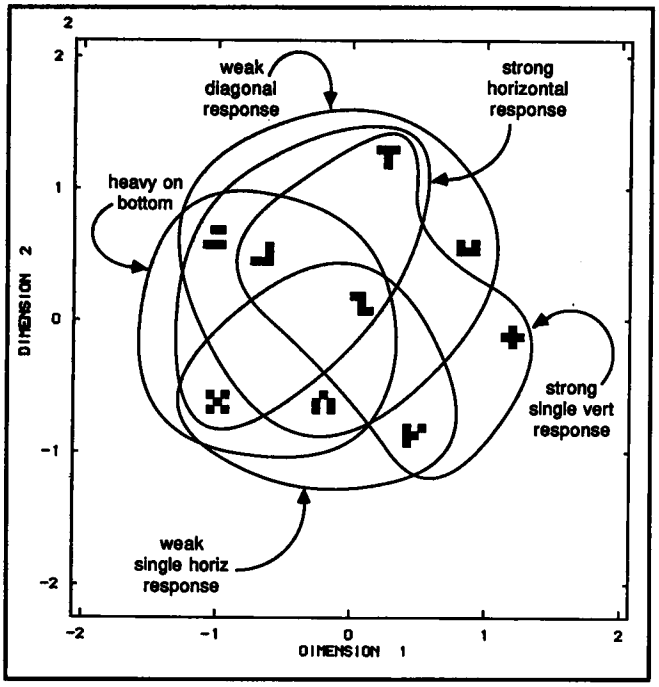

Figure 10. A multidimensional scaling solution for one 14-4 split network.

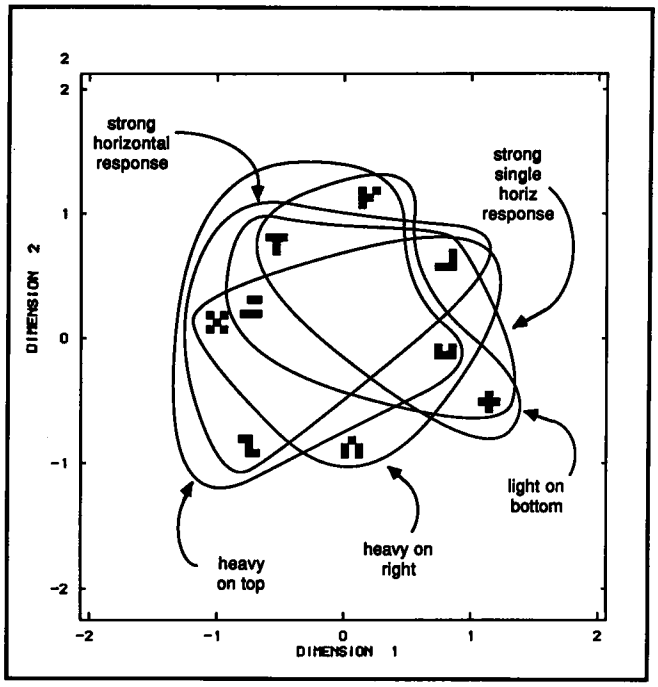

Figure 11. A multidimensional scaling solution for one unsplit network.

and position-independent shape properties. For each comparison, the eight correlation matrices were combined, and this collection of values was correlated with the corresponding similarity measures. As the correlation coefficients displayed in Table 3 reveal, the split networks reflect the position-independent stripe properties more accurately than do the unsplit networks, emphasizing the advantage of the split networks at using position-independent shape properties. In addition, the split networks were also influenced more by the position-dependent properties than were the unsplit networks. These results, then, constitute evidence that the split networks were better able to use all of these properties for shape identification than the unsplit networks.

Because the stripe patterns usually encompass the entire input array, the "what" nodes in the split networks tend to develop position-independent internal representations of shape. An unsplit network, whose hidden nodes tend to have strong connections to both "what" and "where" output nodes, tends to rely on nodes with receptive fields that include contiguous excitatory regions. These nodes are less efficient at identifying shapes, because different sets of hidden nodes must be used to represent a shape when it occurs in different positions. However, these nodes also provide information about location, and can thus contribute to the computation of both identity and location.

Our final analysis measured the relative influences of the "what" and "where" tasks on the split and unsplit models. Each combination of shape and location produces a particular activation pattern across the 18 hidden nodes. If the hidden node representation is strongly influenced by "what," then the activation patterns should be similar whenever a particular shape is present in the input. Similarly, if the hidden node representation is strongly influenced by "where," then the activation patterns should be similar whenever shapes appear at a particular location.

To test the relative influences of "what" and "where," the activation level induced by each combination of shape and location for each hidden node was correlated with the activation level induced by each other combination of shape and location, resulting in an 81 x 81 correlation matrix. The values in this matrix were then correlated with

Table 2

Group	14-4 Split	Unsplit
Horizontal Stripe	8	5
Vertical Stripe	5	4
Diagonal Stripe	5	2
Top Border	0	6
Bottom Border	7	7
Left Border	2	5
Right Border	2	4
Single Horizontal	8	8
Single Vertical	5	5
Single Diagonal	7	3

Number of multidimensional scaling solutions for the "what" output nodes of the unsplit and 14-4 split models that could be grouped by categories defined by stripe and border nodes.

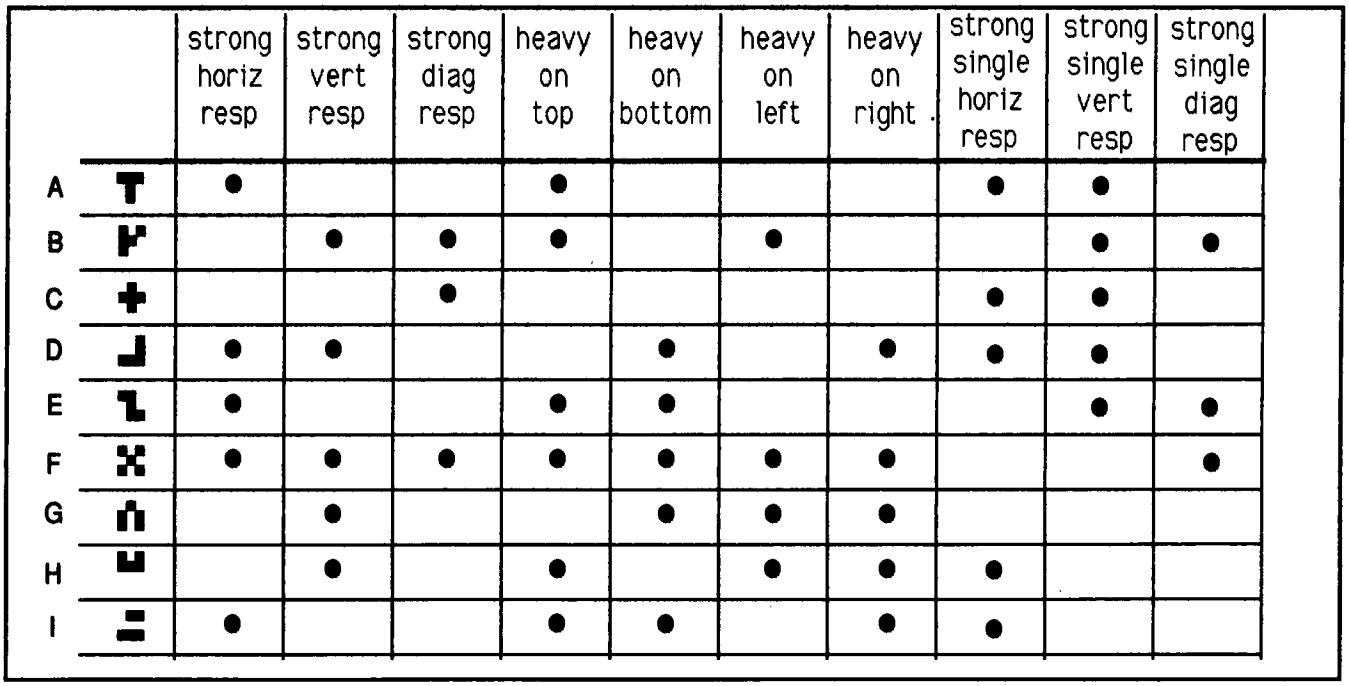

Figure 12. The classification scheme used to group nodes in the multidimensional scaling solutions.

two other matrices. In the first, each shape/location pair in which the shapes matched was coded with a 1, and the rest were coded with a 0. In the other matrix, combinations with matching positions were coded with a 1, and nonmatching positions were coded with a 0. These correlations were done separately for each of the eight 14-4 split and eight unsplit networks. As Table 4 shows, the correlations using position information are higher in the unsplit model than in the split model, $F(1,14) = 20.8$, $p < .001$. Conversely, the correlations using shape information are, if anything, higher in the split model than in the unsplit model, although the difference is small and not significant. The differing roles of "what" and "where" are further documented by the interaction between these factors, $F(1,28) = 22.0$, $p < .001$. In general, location information has a much stronger influence on both models, and that influence is significantly larger in the unsplit model.

Table 3

	TYPE OF SHAPE PROPERTY	
	position-independent	**position-dependent**
Split	0.44	0.30
Unsplit	0.28	0.20
difference	0.16	0.10
$t(3237)$	10.49	6.07
p	<.00005	<.00005

Relative influence of position-independent (stripes) and position-dependent shape properties on hidden-to-output connections. See text for explanation.

Why the Split and Unsplit Models Adopt Different Solutions

Identifying a shape requires that position information be ignored, just as locating an object requires that shape information be ignored. Because these two problems are in this sense independent, it should not be surprising to find that they can more easily be solved by two independent mechanisms. Provided that there were enough processing resources, splitting the system into two independent networks allowed the "what" system to make better use of position-independent feature detectors, and it may have prevented other types of interference as well.

The unsplit network was not constrained to develop in a particular way, and in principle could have developed the same configuration of weights present in the split net-

Table 4

	TASK	
	"what"	**"where"**
Split	0.076	0.494
Unsplit	0.067	0.559

Relative influence of shape information ("what") and location information ("where") on split and unsplit models. See text for explanation.

works. However, as the results from the 9-9 split show, when more nodes are available, the network falls into a less efficient coding scheme for location, using most or all of the available nodes. In particular, during early stages of learning, location information exerts a stronger influence on the weight changes than does identity information (see Figures 6 & 7). Because the "what" and "where" tasks are independent, in many cases the weight changes needed to improve performance on the identification task will conflict with the changes needed to improve performance on the location task. In such cases, the weight changes made will be in the direction that produces the larger decrease in the total error. Typically, this will be in the direction specified by the location computation, because a given weight change tends to produce larger improvements in the performance of the easier computation. Thus, the changes that occur early in the learning sequence will be dominated by the location task, and many hidden nodes will begin to develop connection patterns well-suited to the computation of location. As these patterns develop, it becomes increasingly difficult for the hidden nodes to adjust so that they can respond to shapes. Specifically, it becomes more difficult to develop position-independent feature detectors. Thus, even though the "where" task could easily be solved with just a few hidden nodes, it ends up dominating a large part of the unsplit network, with the consequence that performance on the shape computation remains relatively poor. With time, the unsplit model can learn to identify the shapes, but only within the confines of the constraints placed on the hidden node connections by the location computation.

Although it is of interest to observe the dynamics of learning within the networks, for present purposes the important observations concern differences in the representations that ultimately were formed in split and unsplit networks. The present concern was to observe the ease of representing both identity and location information, and the learning results provide further illumination into how different types of representational capacities depend on the structure of the system. If our analysis is correct, then the advantage that comes from partitioning the system into two independent systems is not specific to the kind of network and learning algorithm used in these simulations. Instead, whenever the system must solve two independent problems and one of the problems is easier than the other, we would expect that dedicating separate resources to each of the problems will improve performance. Even when the problems are equally difficult, dedicating separate resources may prove to be advantageous.

Conclusions

Taken together, the results of Experiments 1 and 2 suggest that partitioning the system into two independent systems results in more efficient computation of identity and location, but only if the hidden nodes are allocated properly between the two systems. Further analyses revealed that the differences in the performance of split and unsplit models arise because the models use different representational schemes to recode the visual input, and that under appropriate conditions the scheme used by the split model is more efficient.

Thus, our results are consistent with the hypothesis that the primate visual system has evolved separate systems for encoding identity and location because it is computationally superior to the single-system approach. Our results also hint at why there is such a great disparity in the sheer amount of neural hardware devoted to the two kinds of processing, with many more cells being devoted to computing object properties in the temporal lobe than are devoted to computing location in the parietal lobe (e.g., Van Essen, 1985). Computing shape is, apparently, a computationally more difficult problem than computing location—even when the shapes vary only in two dimensions. However, in order to evaluate the degree to which the results support these inferences, a number of issues must be addressed.

First, we have inferred that some types of a two-system mechanism are desirable because it is easier to form representations of object identity and location in such systems. However, it is possible that a two-mechanism system would be desirable even if this were not true. That is, although behavior is guided by information about object identity and spatial relations, the two kinds of information tend to have different roles in guiding behavior, and hence would be usefully segregated. Identity information is used primarily to give the system access to previously stored information, which allows the organism to choose the best course of action in the presence of a given object or objects. In contrast, spatial information is used primarily to guide motor movements per se. Thus, because information about object identity and spatial location must be made available to different systems, segregating these types of information within the visual system may be advantageous. If identity and spatial information were tied together in the output of the visual system, a process that is only concerned with one of the kinds of information would need to filter out the extraneous information at what could be considerable computational expense.

Note, however, that the force of this argument is to suggest that the *output* of the "what" and "where" processes should be segregated. This in itself places little constraint on the internal structure of the visual system. Indeed, both the split and unsplit models can be interpreted as satisfying the constraint that the representations of identity and location information be segregated. Thus, although the computational demands of the interface with other cognitive processes may place structural constraints on the visual system, these constraints are not sufficient to explain the relatively early segregation of identity and location processing in that system.

Perhaps the most fundamental objection to our interpretation of the results is that the task performed by the

models was a pale imitation of the task performed by primate visual systems. One important difference is simply the scale of the problem. The simulation models were required to learn to identify a small number of shapes that could occur at a small number of locations, but primates can identify a far greater number of objects that project images on far larger retinae. In addition to scale, the specification of the computation performed by the visual system was also highly simplified. The specification of identity ignored factors such as variations in the orientation and size of the input image, occlusion and stimulus degradation, and variability among the members of a category. Furthermore, we restricted the input to two-dimensional patterns. Similarly, the specification of location involved explicitly representing the "retinal" location of an object, but did not involve combining this information with information about eye, head, or body position in order to determine the egocentric or allocentric position of the object in three dimensions. Finally, the simulations ignored a number of problems faced by the primate visual system, including figure/ground segregation and the processing of color and motion.

Given these differences, one could question whether the present results would be obtained if the simulation models more closely approximated the primate visual system. There are reasons to believe that they would. The results suggest that the two-systems approach can be more efficient because the computation of identity and location place different constraints on the intermediate representations used in the input-output mapping. It seems likely that increasing the scale or the complexity of the problems would amplify the differences in these constraints, thereby producing an even greater advantage for the two-systems mechanism. Furthermore, adding the third dimension would increase the difficulty of computing location only by a relatively small increment (adding one more degree of freedom), compared to an explosion in the number and types of differences in images of shape that would result. Similarly, adding motion information actually seems likely to aid in the computation of location (as well as assisting in figure/ground segregation), but is unlikely to facilitate the identification of shape per se.

The essential observations seem to be that location can vary along fewer dimensions than can object properties, and that the type of dimensions used to represent location are different from the types of dimensions used to represent object properties. Thus, any system that is forced to represent combinations of both types of information in the input is going to be at a disadvantage compared to systems that divide and conquer.

Finally, it could be argued that the results presented here are dependent on the particular architecture and learning rule used in the simulations. That is, although the best split models learned more quickly than the unsplit model in our simulations, this result would not be very interesting if it occurs only with feed-forward networks and the generalized delta rule. Thus, it is important to note that our results do not speak to network design so much as they speak to the inherent structure in certain kinds of information-processing problems. The role of the hidden layer is to recode the input in a way that makes explicit the information needed to respond properly. As was true in Lehky and Sejnowski's (1988) results, ours serve to reflect properties of the stimuli and the task, not the specific computational architecture used. That is, the results described here reflect the nature of the information available to be used in performing computations, which will affect all classes of devices that perform these functions. When object properties and location must be represented explicitly from a single input, the nature of the information to be abstracted is sufficiently different that a properly constructed system will gain considerable advantage if it uses two separate systems.

Method

Operation of the Models

Following Rumelhart et al. (1986), an input was "presented" to the system by setting the activation values of some of the input nodes to 1 and setting the activation of the others to 0. Activation then propagated forward through the system, with the hidden nodes computing their activation values as a function of the input they received from the input nodes, and the output nodes in turn computing their activation values as a function of the input they received from the hidden nodes. The activation rule used to determine the output of each node was the sigmoidal function

$$o_j = \frac{1}{1 + e^{-(\Sigma w_{ji} o_i + \theta_j)}} \qquad \text{(Eq. 1)}$$

where o_j is the activation value of node j, w_{ji} is the strength of the connection from node i to node j, and θ_j is the bias of node j (which is similar in function to a threshold). Given this function, the activation of a node could take on any real value between 0 and 1.

After the output nodes computed their activation values, the generalized delta rule (Rumelhart et al., 1986) was used to modify connection strengths. The application of this rule involved several steps. First, the activation value of each output node was compared to the target value of that node (i.e., the correct response of that node given the input pattern). This comparison resulted in an *error* signal ∂_j for each output node, where ∂_j is given by

$$\partial_{j(\text{output})} = (t_j - o_j) o_j (1 - o_j). \qquad \text{(Eq. 2)}$$

Following Rumelhart et al. (1986) the target values were set at 0.9 and 0.1, rather than 1.0 and 0.0, and appropriate activations beyond these value were treated as errorless responses.

When the error signals were computed, they were propagated back to the hidden nodes, which in turn computed their error signals according to the function

$$\partial_{j(\text{hidden})} = (\Sigma w_{kj} \partial_k)\, o_j (1 - o_j) \qquad \text{(Eq. 3)}$$

where w_{kj} is the weight on the connection from hidden node j to output node k. After the error signals were determined, the strength of each connection was modified according to the equation

$$\Delta w_{ji}(n + 1) = \eta(\partial_j o_j) + \alpha \Delta w_{ji}(n) \qquad \text{(Eq. 4)}$$

where η is the learning rate and n indexes the trial number. Inclusion of the term $\alpha \Delta w_{ji}(n)$ (known as the "momentum") has been shown to increase the rate of learning by modifying the weight changes on a given trial as a function of the weight changes that occurred on other recent trials (Rumelhart et al., 1986). The parameter α controls the degree to which past learning trials will influence learning on the present trial. In the simulations reported here, α was set at .80 and η was set at .75. (In other simulations we varied these values and found no qualitative changes in the pattern of results.)

Procedure

Performance of the split and unsplit models was evaluated by presenting each model with multiple series of trials. On each trial an input was given to the system. The system then determined its output, and the weights on the connections were changed in the manner described above. Trials were grouped into blocks so that in each block each of the nine shapes appeared in each of the nine locations once. The shape/location combinations were randomly ordered within each block.

Each network was simulated in eight separate sets of 300 blocks each. At the beginning of a set of trials each weight was randomly assigned a value between -.3 and .3. Performance on a given trial was measured by summing the squared differences between the activation and target values of each output node. Performance on a block of trials was determined by summing the summed squared error across all the trials within a block

Acknowledgments

This work was supported by AFOSR Grant 88-0012 awarded to the third author, by a Whitaker Health Sciences Graduate Fellowship awarded to the second author, and by a grant from the Alfred P. Sloan Foundation to the M.I.T. Center for Cognitive Science. We thank Kris Kirby for help in developing the simulator. Requests for reprints should be sent to J. G. Rueckl, Department of Psychology, Harvard University, Cambridge, MA 02138 USA.

Note

1. Another effect of the additional connections is related to the magnitude of the weight changes determined by the generalized delta rule. Notice in Equation 4 that the change in the weight of the connection from node i to node j is proportional to the error signal for node j. For the output nodes, the error values for the split and unsplit networks are unlikely to differ systematically early in the learning sequence, and thus the magnitude of the weight changes for the hidden-to-output connections should not differ between the networks. For the hidden nodes, however, the error values for the split and unsplit models are likely to differ systematically early in the learning sequence. This is because the error value for a hidden node is determined by the error propagation scheme given in Equation 3. In this scheme, the error value ∂ is a function of $\Sigma w_{kj} \partial k$, where ∂_k is the error of output node k, and w_{kj} is the weight of the connection from node j to node k. Because each hidden node in the unsplit model is connected to twice as many output nodes as are the hidden nodes in the split model, the error term for hidden nodes is likely to be greater in magnitude in the unsplit model than in the split model. Because the magnitude of the weight changes is proportional to the magnitude of the error terms, each learning trial will tend to produce larger weight changes on the input-to-hidden node connections in the unsplit model. Thus, the unsplit model will tend to take larger steps in weight space, and initial performance will improve more quickly. Nonetheless, the results clearly show that despite this advantage for the unsplit model, other factors are at play, and these other factors confer an advantage onto the split model.

References

Feldman, J. A. (1985). Four frames suffice: A provisional model of vision and space. *The Behavioral and Brain Sciences, 8,* 265–289.

Gross, C. G., & Mishkin, M. (1977). The neural basis of stimulus equivalence across retinal translation. In S. Harnad, R. Doty, J. Jaynes, L. Goldstein, & G. Krauthamer (Eds.), *Lateralization in the Nervous System* (pp. 109–122). New York: Academic Press.

Hinton, G. (1981). A parallel computation that assigns canonical object-based frames of reference. *Proceedings of the Seventh International Joint Conference on Artificial Intelligence.* Vancouver, B. C.

Lehky, S. R., & Sejnowski, T. J. (1988). Neural network model for the cortical representation of surface curvature from images of shaded surfaces. In J. S. Lund (Ed.), *Sensory Processing in the Mammalian Brain.* Oxford: Oxford University Press.

Marr, D. (1982). *Vision.* New York, NY: W. H. Freeman.

McClelland, J. L., & Rumelhart, D. E. (1986). *Parallel Distributed Processing: Explorations in the Microstructure of Cognition. Volume 2: Psychological and Biological Models.* Cambridge, MA: M.I.T. Press.

Rumelhart, D., Hinton, G., & Williams, R. (1986). Learning internal representations by error propagation. In D. E. Rumelhart & J. L. McClelland (Eds.), *Parallel Distributed Processing: Explorations in the Microstructure of Cognition. Volume I: Foundations.* Cambridge, MA: M.I.T. Press.

Rumelhart, D. E., & McClelland, J. L. (1986). *Parallel Distributed Processing: Explorations in the Microstructure of Cognition. Volume I: Foundations.* Cambridge, MA: M.I.T. Press.

Sejnowski, T. J., Koch, C., & Churchland, P. S. (1988). Computational neuroscience. *Science, 241,* 1299–1306.

Ungerleider, L. G. & Mishkin, M. (1982). Two cortical visual systems. In D. J. Ingle, M. A. Goodale, & R. J. W. Mansfield (Eds.), *The Analysis of Visual Behavior.* Cambridge, MA: M.I.T. Press.

Van Essen, D. C. (1985). Functional organization of primate and visual cortex. In A. A. Peters & E. G. Jones (Eds.), *Cerebral Cortex.* New York: Plenum.

Zipser, D. & Andersen, R. A. (1988). A back propagation programmed network that simulates response properties of a subset of parietal neurons. *Nature, 331,* 679–684.

This article has been cited by:

1. Michael C. Mozer, Adrian Fan. 2008. Top-Down modulation of neural responses in visual perception: a computational exploration. *Natural Computing* **7**:1, 45-55. [CrossRef]
2. Raffaele Calabretta, Andrea Di Ferdinando, Domenico Parisi, Frank C. Keil. 2008. How to Learn Multiple TasksHow to Learn Multiple Tasks. *Biological Theory* **3**:1, 30-41. [Abstract] [PDF] [PDF Plus]
3. Günter P. Wagner, Mihaela Pavlicev, James M. Cheverud. 2008. The road to modularity. *Nature Reviews Genetics* **8**:12, 921-931. [CrossRef]
4. Andrea Di Ferdinando, Domenico Parisi, Paolo Bartolomeo. 2007. Modeling Orienting Behavior and Its Disorders with "Ecological" Neural NetworksModeling Orienting Behavior and Its Disorders with "Ecological" Neural Networks. *Journal of Cognitive Neuroscience* **19**:6, 1033-1049. [Abstract] [PDF] [PDF Plus]
5. Markus Knauff, Thomas Fangmeier, Christian C. Ruff, P. N. Johnson-Laird. 2003. Reasoning, Models, and Images: Behavioral Measures and Cortical ActivityReasoning, Models, and Images: Behavioral Measures and Cortical Activity. *Journal of Cognitive Neuroscience* **15**:4, 559-573. [Abstract] [PDF] [PDF Plus]
6. M.M. Poulton, R.A. Birken. 1998. Estimating one-dimensional models from frequency-domain electromagnetic data using modular neural networks. *IEEE Transactions on Geoscience and Remote Sensing* **36**:2, 547-555. [CrossRef]
7. Manfred Spitzer. 1995. *Current Opinion in Psychiatry* **8**:5, 317-329. [CrossRef]
8. William Bechtel. 1993. Currents in connectionism. *Minds and Machines* **3**:2, 125-153. [CrossRef]
9. Robert A. Jacobs, , Michael I. Jordan . 1992. Computational Consequences of a Bias toward Short ConnectionsComputational Consequences of a Bias toward Short Connections. *Journal of Cognitive Neuroscience* **4**:4, 323-336. [Abstract] [PDF] [PDF Plus]
10. Isabelle Otto, Philippe Grandguillaume, Latifa Boutkhil, Yves Burnod, Emmanuel GuigonBurnod. 1992. Direct and Indirect Cooperation between Temporal and Parietal Networks for Invariant Visual RecognitionDirect and Indirect Cooperation between Temporal and Parietal Networks for Invariant Visual Recognition. *Journal of Cognitive Neuroscience* **4**:1, 35-57. [Abstract] [PDF] [PDF Plus]
11. Raymond D. Rimey, Christopher M. Brown. 1991. Controlling eye movements with hidden Markov models. *International Journal of Computer Vision* **7**:1, 47-65. [CrossRef]
12. I. J. Mitchell, J. M. Brotchie, G. D. A. Brown, A. R. Crossman. 1991. Modeling the functional organization of the basal ganglia. A parallel distributed processing approach. *Movement Disorders* **6**:3, 189-204. [CrossRef]
13. Randall C. O'Reilly, Stephen M. Kosslyn, Chad J. Marsolek, Christopher F. Chabris. 1990. Receptive Field Characteristics That Allow Parietal Lobe Neurons to Encode Spatial Properties of Visual Input: A Computational AnalysisReceptive Field Characteristics That Allow Parietal Lobe Neurons to Encode Spatial Properties of Visual Input: A Computational Analysis. *Journal of Cognitive Neuroscience* **2**:2, 141-155. [Abstract] [PDF] [PDF Plus]